Contemporary Engineering Sciences, Vol. 4, 2011, no. 5, 201 – 220

PID Parameter Selection

Based on Iterative Learning Control

M. Rezaei

Kerman, Iran
University of Kerman
Electrical Engineering Department
rezaei.mohsen@hotmail.com

A. Gharaveisi

Kerman, Iran
University of Kerman
Graduate School of Electrical Engineering

A.A. Rezaei

Kerman, Iran
Kerman's Combined Cycle Power Plant
MAPNA Group

Abstract

In this paper, a novel method for designing PID controller is proposed. It uses Iterative Learning Control (ILC) for producing an optimum control signal for the plant and then by using a regression method (Least Squared Error), adjusts PID coefficients so that it acts like the ILC. Then PID is implemented on the plant. This method is simulated on automatic voltage regulating system and results are compared by a PID controller. The results show the effectiveness of this method.

Keywords: Iterative learning control, PID design, regression

INTRODUCTION

Iterative learning control is a control method which was mainly introduced to improve the performance of processes that are repeated periodically over and over. It is used when a precise trajectory tracking is needed, for instance in robotics [Tayebi and Islam, 2006; Wang and Cheah, 1998; Moon, Doh, and Chung, 1997], hard disk position control [Kang and Kim, 2005], electro-pneumatic servo systems [Yu et al., 2004], industrial processes [Pandit and Baque, 1997], injection molding processes, food production facilities, robotic assembly lines, chemical batch reactors [Ratcliffe et al., 2005] and even for density control of freeway traffic flow [Hou, Xu and Yan, 2008]. The idea of ILC algorithm was first introduced by Arimoto and his colleagues [Arimoto, Kawamura, and Miyazaki, 1984]. They showed that in certain conditions, the previous iteration data can be used to improve the current iteration process. In better words, it uses the previous iteration control signal and error signal and tries to make a better control signal for the next iteration and keep error of the next iteration as low as possible. But it should be mentioned that it is not always possible to make current iteration better than previous, because some conditions should hold true, otherwise, whether algorithm dose not converge or makes large transients before convergence.

It can be shown that if conditions are met, after some iterations tracking error in every moment will tend to zero [Arimoto, Kawamura, and Miyazaki, 1984, Yang et al., 2008]. This property is called perfect tracking [Moon, Doh, and Chung, 1997]. For this reason, several algorithms of ILC have been developed for applications of precision motion and tracking control. Many researchers have worked on different ILC algorithms and have shown new applications of it (for a brief history refer to [Tayebi and Islam, 2006]).

Today, ILC is subject of many researches. New methods and algorithms have been designed in order to boost its performance or robustness. But, a drawback in ILC is that it depends highly on plant dynamics. So if a change in plant dynamics occurs, its performance will be disturbed and the algorithm must be run several times again, in order to learn the new dynamics. There are several papers that have shown new ways to make ILC more robust, such as using Q-Filter in. Q-filter is a low-pass filter that can cancel high frequency uncertainties, and consequently improve robustness. But at the other hand, it has an adverse effect on the performance because it affects the high frequency dynamics of the plant itself, which in turn affects the transient manner of the system response. This problem is overcome by using a switched Q-filter with two frequencies, a higher one when better performance needed and a lower frequency when more robustness is required [Bristow, Alleyne and Tharayil, 2007].

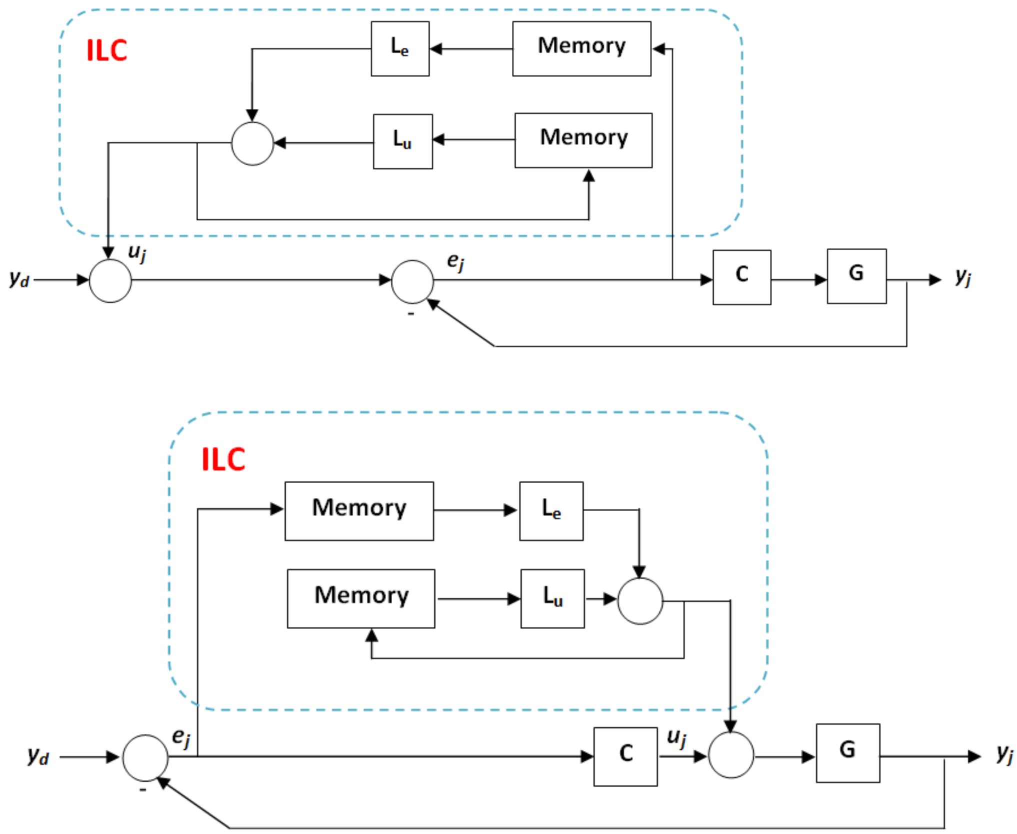


Figure-1, Series (Top) and parallel (bottom) implementation of ILC+PID

In addition to using Q-Filter, another idea is to combine ILC and PID controllers as done by [Gunnarsson and Norrlof, 2001; de Roover and Bosgra, 2000]. Since PID is the most common controller in industry and also has good robustness against uncertainties and disturbances, it's a good idea to combine it with ILC which has good performance. In this manner, the control signal is composed of two signals, one from PID, which improves robustness, and another signal form ILC, which is used to improve the performance. In this case, ILC is a plugged-in controller which acts in parallel or series with the other controller. This kind of combination can be used in two ways: in parallel and series. A schematic of these two kinds of implementation is shown in figure-1.

Some other researchers have used state feedback in addition to ILC for achieving better robustness and tracking error, like [Shi, Gao and Wu, 2005].

The method used in this paper is as follows. It uses PID alone as controller, but the procedure of tuning PID parameters is the result of an ILC algorithm run. Since the final controller is PID, it has a good compromise between performance and robustness. Difference between this method and previously works that used PID and ILC simultaneously, is that the previous works apply ILC as a plug-in controller which in turn needs more hardware and memory to store signals permanently. But in this paper in the final step of design, ILC is totally removed and a conventional PID is replaced with ILC which is more popular and doesn’t need any additional hardware or memory. So, it seems that it is more cost effective and practical.

The algorithm is tested on automatic voltage regulator (AVR) model in Matlab. In a generator, when a voltage drop (or rise) occurs, it is necessary to regulate the voltage fast and precise. Because of high precision needed, it seems that ILC can be a good choice as AVR control law and for gaining robustness the ILC controller is replaced with a PID. AVR model used in this paper is extracted from reference [Naderi, Gharaveisi and Rashidinejad, 2007].

There may be an argument that why ILC is used on AVR system which is not in nature a repetitive process? Applications of ILC in non-repetitive processes have been discussed already in references like [Velthuis, 2000; Dixon et al., 2002; de Vries and Velthuis; Ruan, Bien and Park, 2008; Iftime and Verhaegen, 2007]. In our case, since we use a software model of real AVR plant, we can run it as much as we wish. Then after designing PID, It can be tested on real system. It means that in some plants that are not repetitive, an offline repetitive design can be done on their approximate model and after PID design, it can be tested for verifying its functionality. A little difference between real plant and its modeled dynamics is not significant because PID can compensate for it. So, after tuning PID it can even be used for non-repetitive tracking.

This paper is organized as follows: ILC problem definition and some important theorems about optimal design of it are stated in section 2. Then, least square error regression method is discussed in section 3. Design algorithm is presented in section 4. AVR model is briefly discussed in section 5 and simulation results are shown in section 6 and are compared with a Ziegler-Nichols PID. Finally, section 7 presents the conclusions.

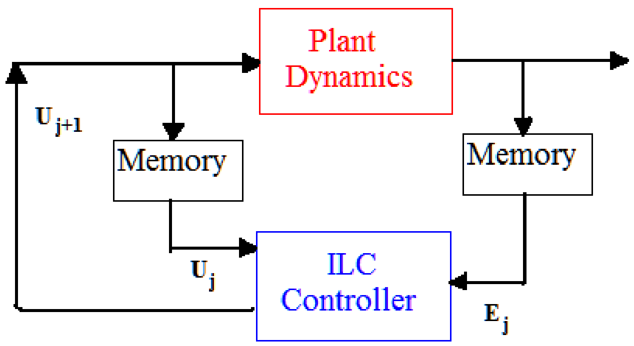


Figure-2, ILC Controller

1. Iterative Learning Control

ILC algorithm is usually formulated for discrete systems. We also use a discrete model here to describe the plant:

$$Y_j = PU_j + d \qquad (1)$$

where:

$$Y_j = \begin{bmatrix} y_j(1) \\ \vdots \\ y_j(N) \end{bmatrix}, U_j = \begin{bmatrix} u_j(0) \\ \vdots \\ u_j(N-1) \end{bmatrix}, d = \begin{bmatrix} d(0) \\ \vdots \\ d(N-1) \end{bmatrix} \qquad (2)$$

This is called a lifted system representation. $y_j(k)$ shows the output in iteration "j" and "k"th sample. $u_j(k)$ is the input to the system, d is disturbance and P is the static map that maps input U to output Y [Tousain, van der Mech´e and Bosgra, 2001]. It can be shown that P is matrix of markov parameters of the system:

$$P = \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & 0 & \vdots \\ \vdots & \vdots & \ddots & 0 \\ CA^{N-1}B & CA^{N-2}B & \cdots & CB \end{bmatrix} \qquad (3)$$

where: A, B and C are the main system describing matrices:

$$\begin{cases} X_j(k+1) = AX_j(k) + BU_j(k) \\ Y_j(k) = CX_j(k) + d \end{cases} \quad k \in [0, N] \tag{4}$$

Markov parameters also can be obtained from the impulse response of the plant. ILC update law is defined as below:

$$U_{j+1} = L_u U_j + L_e E_j \tag{5}$$

where:

$$E_j = Y_d - Y_j \tag{6}$$

L_u and L_e are called Learning Gain Matrices. Y_d is the desired trajectory. If we choose learning gains properly, algorithm will converge. It is stated in the following theorem. A schematic of ILC controller is shown in figure-2.

Theorem1: For plant and ILC update law below:

$$\begin{cases} Y_j = PU_j + d \\ U_{j+1} = L_u U_j + L_e E_j \end{cases} \tag{7}$$

if:

$$\max_j(\lambda_j(L_u - L_e.P)) < 1 \tag{8}$$

Then the algorithm asymptotically converges, and control and error vector will converge to fixed vectors U_∞ and E_∞ such that:

$$\begin{cases} U_\infty = (I - L_u + L_e P)^{-1} L_e Y_d \\ E_\infty = (I - P(I - L_u + L_e P)^{-1} L_e) Y_d \end{cases} \tag{9}$$

[Moore , Chen and Ahn, 2006].

Note that if $L_u = I$, error will converge to zero. Then if we choose L_e a diagonal matrix like this:

$$L_u = \begin{bmatrix} \gamma & 0 & \cdots 0 \\ 0 & \gamma & \cdots 0 \\ \vdots & & \ddots \\ 0 & 0 & \cdots \gamma \end{bmatrix} \tag{10}$$

the convergence conditions will be simply as:

$$|1 - p_0.\gamma| < 1 \tag{11}$$

where $p_0 = CB$ is the first markov parameter. This is a simple way of designing an ILC controller which is called: P-Type Learning. The only thing we need from plant is one parameter (p_0). But the problem is that in many plants the first markov parameter is zero and so this method doesn't guarantee convergence. The solution is choosing other methods. The following theorem helps to solve this problem.

Theorem2: Suppose the following cost function:

$$J_{j+1} = E_{j+1}^T Q E_{j+1} + U_{j+1}^T R U_{j+1} + \delta U_{j+1}^T S \delta U_{j+1} \tag{12}$$

where: $\delta U_{j+1} = U_{j+1} - U_j$ and Q, R and S are positive semi-definite weighing matrices . Then L_u and L_e matrices that make this cost function minimum are:

$$\begin{cases} L_u = (P^T QP + R + S)^{-1}(P^T QP + S) \\ L_e = (P^T QP + R + S)^{-1} P^T Q \end{cases} \tag{13}$$

and error will monotonically asymptotically converge to:

$$E_\infty = (I - P(P^T QP + R)^{-1} P^T Q)(Y_d - d) \tag{14}$$

This is the basis of LQ Optimal Design of ILC. The role of Q matrix is weighing error. It will cause the error to be low enough. R is a weighting matrix that makes the control signal of ILC low so that saturation doesn't happen. S matrix ensures that we have a smooth change between iterations and as a result, big peeks in error don't occur. In fact S is related with the convergence speed.

2. LEAST SQUARED ERROR REGRESSION

This method is used when there are input and output data of a plant and we want to find a model for that system so that the squared error between real plant output and the model output is minimum. The method presented here is discussed in many text related to system identification or adaptive control, such as [Astrom, Wittenmark, 1989]. The method is discussed below briefly for the PID case, but is general. The aim of regression is to replace input-output data of ILC by a PID controller with three parameters which will yield minimum error between them.

Suppose standard PID transfer function:

$$U_1(s) = \left(K_p + \frac{K_i}{s} + K_d s\right) E(s) \tag{15}$$

Where: $E(s)$ is error and $U_1(s)$ is PID output. In time domain:

$$u_1(t) = K_p e(t) + K_i \int_{-\infty}^{t} e(\tau) d\tau + K_d \dot{e}(t) \tag{16}$$

By sampling that equation in sampling time T_s :

$$u_1(kT_s) = K_p e(kT_s) + K_i \int_{-\infty}^{kT_s} e(\tau) d\tau + K_d \dot{e}(kT_s) \tag{17}$$

kT_s will be abbreviated as k since now. If there are N samples, the above equation can be written as a matrix form like below:

$$U_1 = \phi\theta \tag{18}$$

Where:

$$U_1 = \begin{bmatrix} u_1(1) \\ \vdots \\ u_1(N) \end{bmatrix}, \phi = \begin{bmatrix} e(1) & \int_{-\infty}^{T_s} e(\tau) d\tau & \dot{e}(1) \\ \vdots & \vdots & \vdots \\ e(N) & \int_{-\infty}^{NT_s} e(\tau) d\tau & \dot{e}(N) \end{bmatrix}, \theta = \begin{bmatrix} K_p \\ K_i \\ K_d \end{bmatrix} \tag{19}$$

Suppose kth line of ϕ is shown by: $\varphi(k)$, then we have:

$$u_1(k) = \varphi(k)\theta = \begin{bmatrix} e(k) & \int_{-\infty}^{kT_s} e(\tau) d\tau & \dot{e}(k) \end{bmatrix} \begin{bmatrix} K_p \\ K_i \\ K_d \end{bmatrix} \tag{20}$$

The aim of regression is to minimize difference between U_1 and U, namely PID and ILC outputs. this function can be used to express the sum of squared error between them:

$$V(\theta) = \frac{1}{2}\sum_{k=1}^{N}(u(k) - \varphi(k)\theta)^2 \tag{21}$$

$u(k)$ is real controller output and $\varphi(k)\theta$ is an estimate for it.

Theorem3: If θ is chosen so that:

$$\phi^T\phi\hat{\theta} = \phi^T U \tag{22}$$

then $V(\hat{\theta})$ is minimum [Astrom, Wittenmark, 1989].

Note that if $\phi^T\phi$ is nonsingular, then θ optimal ($\hat{\theta}$) can be calculated from below:

$$\hat{\theta} = (\phi^T\phi)^{-1}\phi^T U \tag{23}$$

Proof: The loss function (22) can be written as:

$$\begin{aligned} 2V(\theta) &= (U - \phi\theta)^T(U - \phi\theta) \\ &= U^T U - U^T\phi\theta - \theta^T\phi^T U + \theta^T\phi^T\phi\theta \end{aligned} \tag{24}$$

The above equation can be written as:

$$\begin{aligned} 2V(\theta) = U^T U - U^T\phi\theta - \theta^T\phi^T U + \theta^T\phi^T\phi\theta + \\ U^T\phi(\phi^T\phi)^{-1}\phi^T U - U^T\phi(\phi^T\phi)^{-1}\phi^T U \end{aligned} \tag{25}$$

Rewriting this equation yields:

$$\begin{aligned} 2V(\theta) = U^T(I - \phi(\phi^T\phi)^{-1}\phi^T)U + \\ (\theta - (\phi^T\phi)^{-1}\phi^T U)^{-1}\phi^T\phi(\theta - (\phi^T\phi)^{-1}\phi^T U) \end{aligned} \tag{26}$$

The first term in (26) is independent of θ, since $\phi^T\phi$ is always nonnegative, $V(\theta)$ has a minimum and is obtained when:

$$\theta = \hat{\theta} = (\phi^T\phi)^{-1}\phi^T U \tag{27}$$

And the theorem is proven.

3. OPTIMAL PID DESIGN

The contribution of this paper is to replace the ILC controller with a PID controller which as close as possible, acts like the ILC controller. When replacement is done, the robustness issue is solved, because PID has good robustness against uncertainties, disturbances and noises. As can be seen in figure-3, the procedure of designing PID is as follows:

1- By using system identification methods, find a markov matrix of the plant.

2- By choosing a proper performance index like J, design proper learning gains L_u and L_e.

3- In order to achieve optimal control vector U_∞ and error vector E_∞ , run the ILC algorithm as much as needed, until it converges.

4- Give E_∞ , its integral and its derivative as inputs and U_∞ as output to the regression unit, to achieve PID parameters.

5- Test PID on the plant and if necessary, adjust it near designed parameters.

These processes are simple to implement and most of them can be done by computer, not directly on the plant.

4. AVR SYSTEM

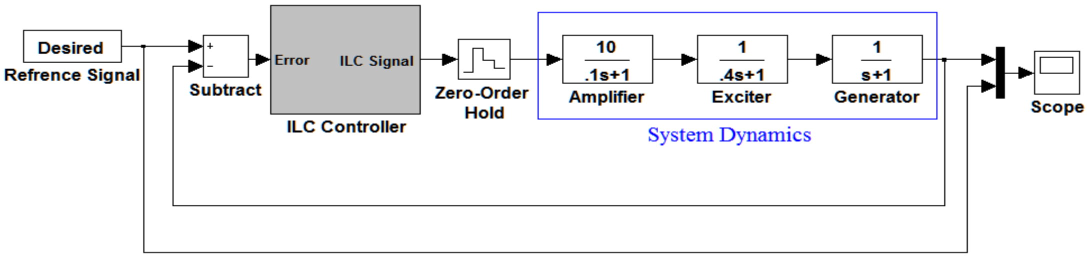


Figure-4, ILC controller for AVR system

In a generator, it is necessary to keep the output voltage as constantly as possible. There are many disturbances in a power system, like “temperature rise, speed change, load change and power factor change,” which all affect the voltage level of the generator [Htay and San Win, 2008]. So, it’s necessary to keep the voltage level constant.

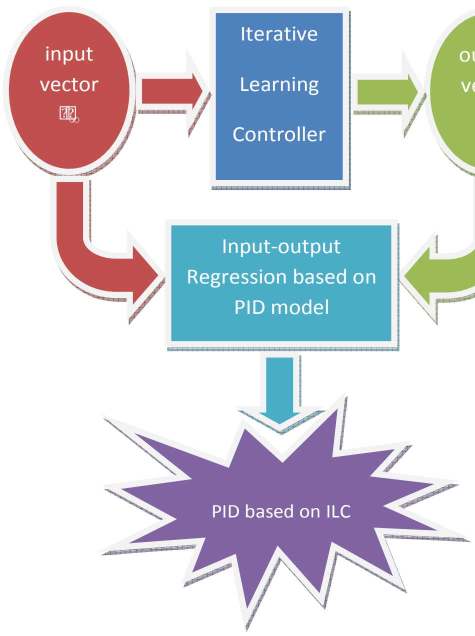


Figure-3, PID design by regression method

In response to active power changes, the input fuel to the turbine (steam, water) must be increased to match the demanded power, or the frequency of the network decreases. This can be done automatically by a system called: “Automatic Generation Control” or AGC. In addition to this, variation in reactive power may change the voltage level. So the exciter should be regulated in order to match the voltage drop(or rise). There must be a voltage regulator device in order to adjust the voltage according to the new conditions. Voltage regulator can be controlled automatically or manually by tap-changing switches, a variable auto transformer and also an induction regulator. When controlling manually, an operator reads the voltage by a voltmeter and decides what to do, but it’s not possible always, especially in modern large networks. AVR system is designed for this purpose. The main duty of AVR, in a power planet, is to maintain generator voltage automatically [F. Naderi, A.A. Gharaveisi and M. Rashidinejad, 2007] which affects the security of the system. But as discussed in ref. [Myinzu Htay and Kyaw San Win, 2008], in general “there are 3 important tasks for AVR system:

1-Better voltage regulation,

2- Stability improvement,

3- Reduce over-voltage on loss of load”.

AVR circuit senses voltage changes and automatically adjusts the exciter field for cope with new conditions by changing its output, which is a setpoint for the plant.

The plant consists of four main parts: Amplifier, Exciter, Generator and Sensors as indicated in [F. Naderi, A.A. Gharaveisi and M. Rashidinejad, 2007]. A schematic of this system is drawn in figure-4.

Amplifier: It has a simple first order model like this:

$$\frac{V_R(s)}{V_e(s)} = \frac{K_A}{1+\tau_A s} \tag{28}$$

with K_A ranging from 10 to 400 and τ_A ranging from 0.02 to 0.1s.

Exciter: It is proposed as:

$$\frac{V_F(s)}{V_R(s)} = \frac{K_E}{1+\tau_E s} \tag{29}$$

K_E ranging from 10 to 400 and τ_E ranging from 0.5 to 0.1s.

Generator: Its model is considered as:

$$\frac{V_t(s)}{V_F(s)} = \frac{K_G}{1+\tau_G s} \tag{30}$$

K_G ranges between 0.7 and 1.0 an τ_G is from 1.0 to 2.0s (full load and no load).

Sensors: Although sensors have a first order dynamic too, we consider them as a unit gain and without dynamics, because they are fast response.

5. SIMULATIONS

As AVR Controller, we can use classic controllers like PID or other methods. Several papers have used new methods for doing so and have obtained better results.

The Idea in this paper, as discussed before, is to use an ILC controller which iteratively boosts its control signal. Then ILC is then replaced by a usual PID controller, which its coefficients are adjusted such that acts like ILC. Some simulations are performed to show the advantages of this plan.

At first, an ILC algorithm is designed for tracking the unit step command. The Results are shown in figures 6 and 7. As can be seen, good tracking is achieved.

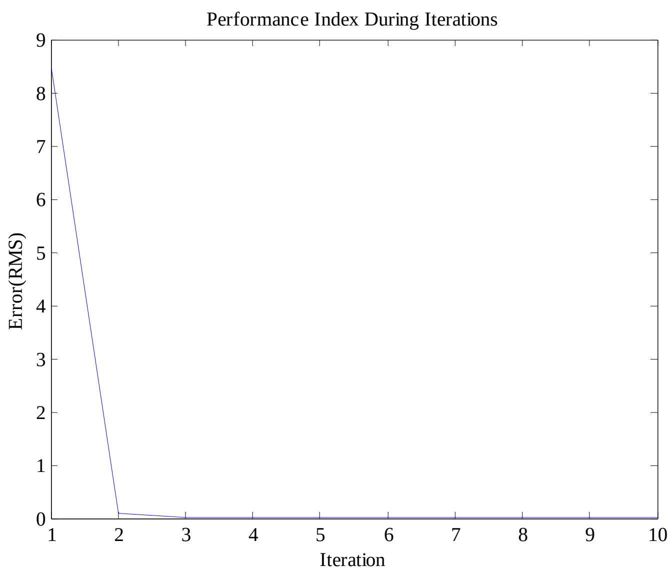


Figure-5, ILC Convergence

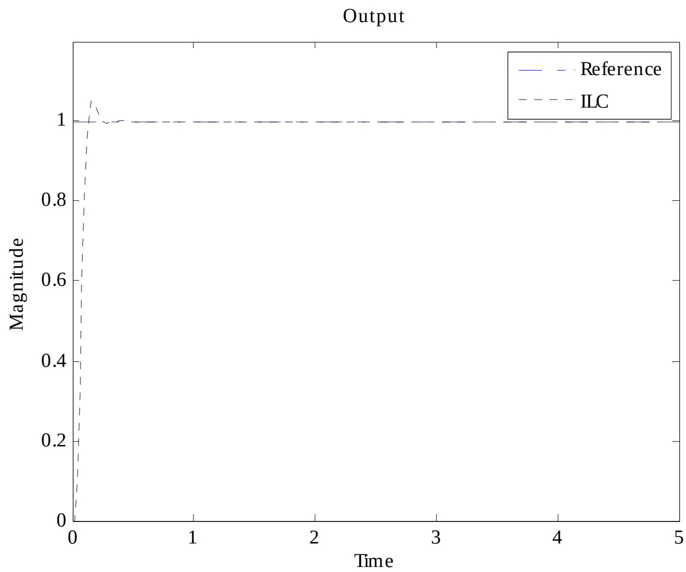


Figure-6, ILC Output

Design parameters for this simulation are: $Q = I, R = 0.01I, S = 0.00001I$.

After this step, a regression is made and ILC controller is replaced by a PID controller. The details of this step were discussed in section 3. After the coefficients were obtained by regression, a simulation was made and the results are presented in figures 5 and 6. From fig. 5 it is seen that the outputs are as closely as possible. Integral of Squared Error for this simulation is about 0.009, which proves that regression is done well. PID parameters produced by regression are:

$P = 0.64342, I = 1.3975, D = 0.028214.$

The Output has a low overshoot and small rise time and settling time. So this PID has good performance and it is expected that it acts to somewhat better than similar PIDs. The next simulation shows this, in comparison with a ZN PID controller.

In the next step, a simulation is made for disturbance rejection and is compared with a Ziegler-Nichols PID. Results are shown in the table. 1. Output graphs are in figures 8 and 9. Results show that the proposed method acts in almost all criteria better than Ziegler Nicholes. The only place where ZN acts better is in rise time,

	ZN PID	Optimal PID
Output Overshoot	19.9 %	14.9 %
Rise Time	0.08 s	0.14 s
Settling Time	2.49 s	0.52 s
Maximum Control Effort	2.42	0.93

Table-1, ZN and optimal PID comparison

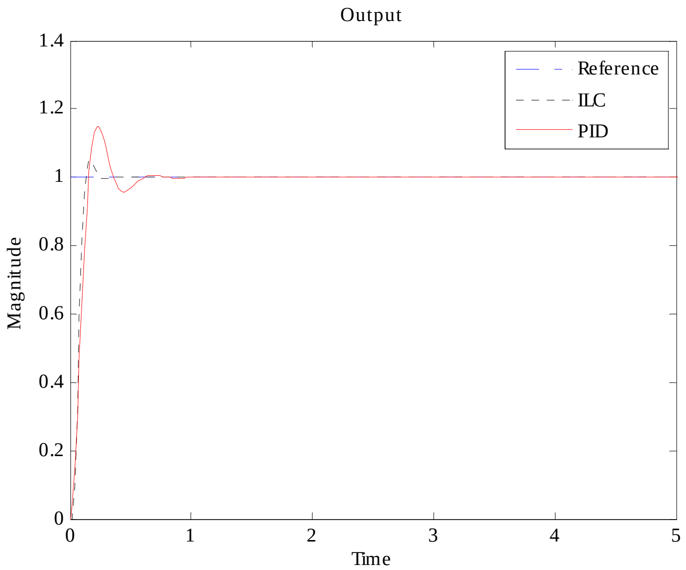
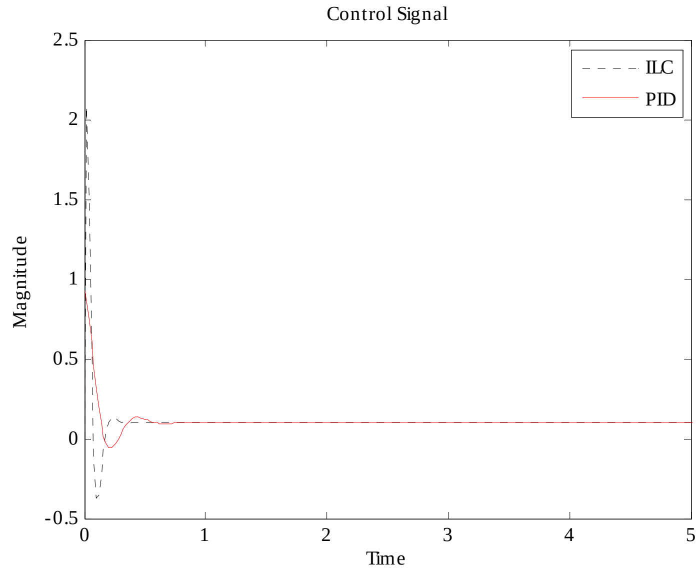


Figure-7, ILC and PID Outputs

which in AVR system is not as important as settling time. Because in AVR it is important that as soon as possible, oscillations are damped (settling time). In this criteria, optimal PID acts about 5 times better than ZN, with a maximum control signal about 2.5 times lower than ZN.

A comparison between two methods is made and the results are shown in table-1. Thus, the algorithm produces acceptable performance criteria.

A



n

Figure-8, ILC and PID Control Signals

ot

her simulation was done in order to test disturbance rejection capability of the designed PID. The results are shown in figure 9 and 10. Disturbance rejection is a key factor in designing industrial controllers, because in real conditions always noise

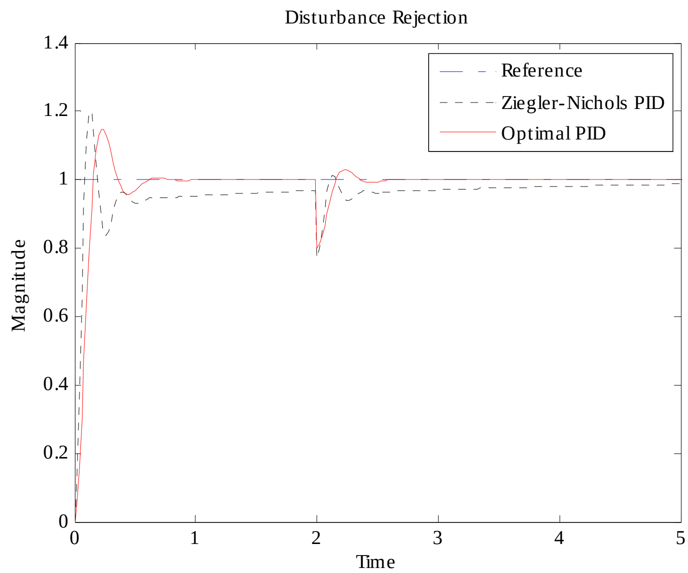


Figure-9, Disturbance Rejection

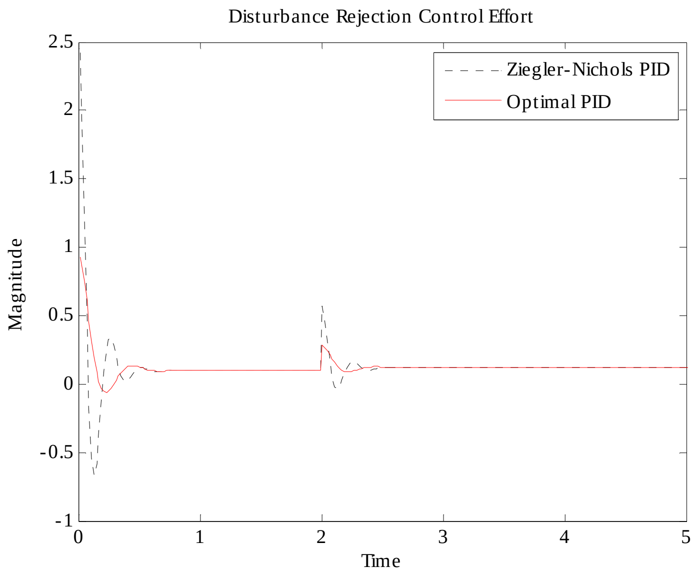


Figure-10, Control Signals

and sudden disturbances are present.

Finally, simulations are made for setpoint change tracking, which results are shown in figures 11 and 12. PID acts better again with lower control signal. It should

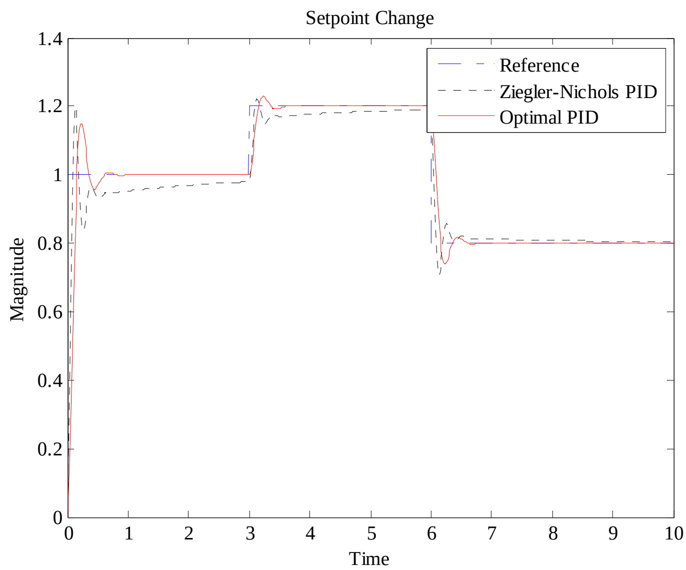


Figure-11, Setpoint Changes

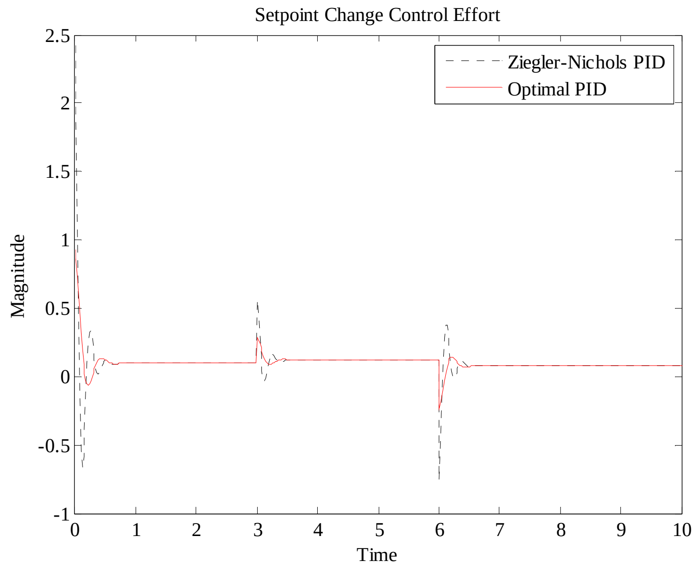


Figure-12, Control Signals

be noted that ILC algorithm can track setpoint changes that repeat in each iterations and cannot response to sudden setpoint changes (or disturbances) in the current iteration. Because, in the time domain, ILC is an open loop controller, in nature. But, by replacing with PID, these problems can be overcome and good robustness is achieved.

6. CONCLUSIONS

A new method for designing PID controller was proposed. It uses ILC for a first design step and then it is replaced by a usual PID controller, whose coefficients are adjusted by a regression method. The aim of regression is to make PID act like ILC, as closely as possible. Then simulations run for setpoint tracking, disturbance rejection and setpoint change, and results were compared with standard Ziegler Nichols PID. In almost all cases, proposed method acted better than ZN PID. The benefit of this method over ILC is simplicity (PID has only three parameters), popularity of PID and good robustness. The results obtained, show effectiveness of this method.

REFERENCES

[1] S. Arimoto, S. Kawamura and F. Miyazaki, Bettering operation of dynamic systems by learning: A new control theory for servomechanism or mechatronic systems, In Proceedings of 23rd Conference on Decision and Control, 1064–1069, Las Vegas, Nevada, December 1984.

[2] K.J. Astrom, B. Wittenmark, Adaptive Control, Prentice Hall, Second Edition, 1989.

[3] D.A. Bristow, A. Alleyne and M. Tharayil, A Time Varying Q-Filter Design for Iterative Learning Control, Proceeding of the 2007 American Control Conference, Marriotte Marquis Hotel at Times Square, New York City, USA, July 11-13, 2007.

[4] D. de Roover, and O.H. Bosgra, Synthesis of Robust Multivariable Iterative Learning Controllers With Application to a Wafer Stage Motion System, International Journal of Control, Vol. 73, No. 10, 968-979 (2000).

[5] T.J.A. de Vries and W.J.R. Velthuis, Toward exploiting the benefits of ILC in non-repetitive motion applications, University of Twente, Netherlands.

[6] W. E. Dixon, E. Zergeroglu, D.M. Dawson, and B.T. Costic, Repetitive Learning Control: A Lyapunov-Based Approach, IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS-PART B: CYBERNETICS, Vol. 32, No. 4, August 2002.

[7] S. Gunnarsson, and M. Norrlof, On the Design of ILC Algorithms Using Optimization, Automatica, Vol. 37, No. 12, 2011-2016 (2001).

[8] Z. Hou, J.X. Xu and J. Yan, An iterative learning approach for density control of freeway traffic flow via ramp metering, Transportation Research Part C: Emerging Technologies, Volume 16, Issue 1, 71-97, February 2008.

[9] M. Htay and K. San Win, Design and Construction of Automatic Voltage Regulator for Diesel Engine Type Stand-alone Synchronous Generator, World Academy of Science, Engineering and Technology 42 (2008).

[10] O.V. Iftime and M. Verhaegen, Model-based learning control with non-repetitive initial conditions, International Journal of Intelligent Systems Technologies and Applications, Volume 2, Number 2-3, 161-173 (2007).

[11] C.I. Kang and C.H. Kim, An Iterative Learning Approach to Compensation for the Servo Track Writing Error in High Track Density Disk Drives, Microsystems Technology, 11 (2005), 623–637.

[12] J.H. Moon, T.Y. Doh, and M.J. Chung, An Iterative Learning Control Scheme for Manipulators, Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robots and Systems, 1997.

[13] K.L. Moore, Y.Q. Chen and H.S. Ahn, Iterative Learning Control: A Tutorial and Big Picture View, 45th IEEE Conference on Decision and Control, San Diego, CA, USA, December 2006.

[14] F. Naderi, A.A. Gharaveisi and M. Rashidinejad, Optimal Design of Type_1 TSK Fuzzy Controller Using GRLA for AVR system, Large Engineering Systems Conference on Power Engineering, 106 - 111 (2007).

[15] M. Pandit and S. Baque, Learning Control of Cyclic Production processes, 6th International Conference on Emerging Technologies and Factory Automation Proceedings, Los Angeles, California, September 9-12, 1997.

[16] J. Ratcliffe, P. Lewin, Eric Rogers, Jari Hätönen, Thomas Harte and David Owens, Measuring the Performance of Iterative Learning Control Systems, Proceedings of the 2005 IEEE International Symposium on Intelligent Control, Limassol, Cyprus, June 27-29, 2005.

[17] X.E. Ruan, Z. Bien and K.H. Park, Decentralized iterative learning control to large-scale industrial processes for non-repetitive trajectories tracking, IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans,Vol. 38, No. 1, 238-252 (2008).

[18] J. Shi, F. Gao and T.J. Wu, Robust design of integrated feedback and iterative learning control of a batch process based on a 2D Roesser system, Journal of Process Control 15, 907–924 (2005).

[19] A. Tayebi and S. Islam, Adaptive Iterative Learning Control for Robot Manipulators: Experimental results, Control Engineering Practice, 14 (2006), 843–851.

[20] R. Tousain, E. van der Mech´e and O. Bosgra, Design strategies for iterative learning control based on optimal control, Selected Topics in Signals, Systems and Control, Vol. 12, September 2001.

[21] W.J.R. Velthuis, Learning Feed-Forward Control, Theory, Design and Applications, PhD Thesis, University of Twente, Enschhede, Netherlands, 2000.

[22] D. Wang and C.C. Cheah, An Iterative Learning Control Scheme for Impedance Control of Robotic Manipulators, The International Journal of Robotics Research, 1998.

[23] S. Yang, Z. Qu, X. Fan and X. Nian, Novel iterative learning controls for linear discrete-time systems based on a performance index over iterations, Automatica 44, 1366–1372 (2008).

[24] S.J. Yu, X.D. Qi, R.C. Han and F. Pan, Practical Design of an Iterative Learning-Sliding Mode Controller for Electro-Pneumatic, International Journal of Information Technology, Vol. 11 No. 5 (2004).

[25] J.G. Ziegler and N.B. Nichols, Optimum Setting for Automatic Controllers, Transactions of ASME, 759-768 (1942).

Received: February, 2011